Analysis of GNAC Volleyball using the Bradley-Terry Model

By
Daniel Karwoski

A Project Submitted in Partial Fulfillment of the Requirements
for the Degree of
Master of Science
in
Statistics

University of Alaska Fairbanks
May 2020

APPROVED:

Margaret Short, Committee Co-Chair
Scott Goddard, Committee Co-Chair
Julie McIntyre, Committee Member
Ron Barry, Committee Member
Leah Berman, Chair
Department of Mathematics and Statistics

Abstract

Ranking is the process by which a set of objects is assigned a linear ordering based on some property that they possess. Not surprisingly, there are many different methods of ranking used in a wide array of diverse applications; ranking plays a vital role in sports analysis, preference testing, search engine optimization, psychological research, and many other areas. One of the more popular ranking models is Bradley-Terry, which is a type of aggregation ranking that has been used mostly within the realm of sports. Bradley-Terry uses the outcome of individual matchups (paired-comparisons) to create rankings using maximum-likelihood estimation.

This project aims to briefly examine the motivation for modeling sporting events, review the history of ranking and aggregation-ranking, communicate the mathematical theory behind the Bradley-Terry model, and apply the model to a novel volleyball dataset.

Introduction

In general, sports modeling attempts to accomplish two things: generate quality predictions for the outcome of future sporting events and determine the factors which are most relevant for success in a contest. To construct a model that makes accurate predictions, one must understand the factors that are most important in the data generating process [5]. Random effects are commonly referred to as "chance" in the context of sporting events; these are unmodeled variables that have an impact on the process outcome. Random effects are present in every contest, but the extent of their influence on the results is generally not consistent between types of games (e.g., a chess match is a contest whose outcome is based almost entirely on skill, and the lottery is a contest that depends solely on chance). It is not surprising that the results of most sporting events are dependent on both skill and luck. Psychological research has found that part of the draw of sports is the excitement that this element of unpredictability brings to competitions [1].

One of the models used to describe sports is the Bradley-Terry model. The Bradley-Terry model was created in the 1920s to rank chess players based on their performance in pairwise matchups. In general, a matchup is an event in which two different choices present themselves. The event results in one of the alternatives "beating" or being "preferred" over the other. Although the model is usually applied to sports, it is also applicable to other areas like preference-testing. The most basic forms of the model rely on the dubious (but practical and not particularly pathological) assumption that contests are independent in order to estimate parameters using maximum-likelihood. The standard model only deals with binary outcomes, but modifications can be made to the model to allow for ties.

Volleyball is considered by many casual fans and sports researchers to be one of the "skill sports" because, in volleyball, almost all gameplay is relevant to the outcome; a chain of many events must occur for a team to win a volleyball match (25 points are required to win a set, and 3 sets are required to win a match). In other words, it is highly improbable for a strong team to be defeated by a weak team based on chance [1]. Consequently, volleyball is more likely to follow predictable patterns, and, presumably, be amenable to modeling. These observations played a significant role in the decision to use volleyball to explore the capabilities of the Bradley-Terry model in this paper.

The data for this project consists of box scores for the in-conference games of Great Northwest Athletic Conference (GNAC) Women's Volleyball during the period between 2016 and 2018, which were scraped from the internet using python's pandas [6] library. All in-conference games, except games that took place on neutral ground, were included in the dataset. The first two seasons served as the training data, while the last season was held out as test data.

This paper will discuss the history of rank aggregation and the Bradley-Terry Model, the specification and theory of Bradley-Terry Models, project data collection methodology, model results, and recommendations for future work, in that order.

1. Ranking and Paired Comparison

1.1 Rank Aggregation

Rank aggregation is the process by which several independent preference rankings, which can be partial or full rankings, about a common set of choices, are merged into an overall ranking [4]. The problem of rank aggregation dates back to the 18th century when Condorcet and Borda studied it in the context of voting systems. More recently, rank aggregation theory has found applications in search algorithms, database systems, bioinformatics, and many other areas [4].

1.2 Paired Comparison

There are numerous ways to aggregate rankings, which range in complexity from simple voting methods like Borda Count, which allots points to candidates according to the order in which each respondent ranks them, to sophisticated Bayesian methods [4]. The method of paired comparisons, which assigns a binary response variable to a comparison between two alternatives, is one of the most frequently used of these variations. One desirable property of paired comparison testing is that it is straightforward to implement and analyze ("Given two, choose one"). In addition to being easy to carry out, this method is easy to interpret, especially when contrasted with studies that require test subjects to rank an entire list of objects or to assign objects ratings on a points scale, one-at-a-time.

The notion of paired-comparison testing has its roots in psychology, where it was first studied as the "Law of Comparative Judgement" by Thurstone in 1927. Thurstone used pairwise comparisons as a way of ranking perceptions about certain stimuli [3]. In 1929, Ernst Zermelo used a similar procedure to create chess rankings. His method drew inferences about competitor abilities based on multiple matches between numerous players. He modeled each game as a Bernoulli-trial (binary outcome), which he assumed to be a function of the players' latent strengths. Then, he aggregated the results to form a generalized Binomial likelihood distribution used to calculate MLE estimates for the strength of each player. This model provides not only a ranking of competitors but also an estimate of the probability of a given competitor defeating any of the other competitors. Zermelo's model rose to prominence as "The Bradley-Terry Model" in 1952 after being rediscovered by R.A. Bradley and M.E. Terry [3].

2. Specification of The Bradley-Terry

The Bradley-Terry model uses the observed values of contest outcomes to predict latent strength or ability scores. As previously stated, these strength scores not only predict contest outcomes but also establish rankings. There are several variations of Bradley-Terry models; the standard model and two models which include home-field advantage will be covered here.

2.1 Standard Model

Suppose there is a tournament with K teams. Let i and j be any two teams in the league competing in a contest:

$$(i, j \in \{1, 2, ..., K\}; i \neq j)$$

Assign each team a real, positive-valued strength parameter which indicates the team's level of ability or skill:

$$\{\pi_1, \pi_2 ... \pi_K\} \in \mathbb{R}^+$$

and log strength :

$$\beta_i = log(\pi_i),\ i = 1, 2, ..., K$$

Note that these parameters are not the realization of a strength distribution(s), but rather represent an innate, unchanging characteristic of the competitors. The Bradley-Terry model defines the log-odds that team i defeats team j as the difference in their log-strengths, $\beta_i - \beta_j \in \mathbb{R}^+$, where outcomes are modeled as independent Bernoulli(P_{ij}) random variables with a binary result 1 in the event that i beats j and 0 otherwise.

The model expresses the probability that i beats j, P_{ij} or $Pr(i > j)$, as the ratio of the strength of team i to the sum of the strengths of team i and team j:

$$P_{ij} = \frac{\pi_i}{\pi_i + \pi_j}$$

This specification implies that solutions are only identifiable up to a constant multiple. This issue is usually dealt with by setting one of the $\pi_i = 1$ for some i or imposing the constraint $\sum \pi_i = 1$.

In the standard Bradley-Terry model where ties are disallowed, the log odds of P_{ij} are equal to the difference in the log strengths of i and j:

$$\text{Log-odds}=\log\left(\frac{P_{ij}}{1-P_{ij}}\right)=\log\left(\frac{P_{ij}}{Pji}\right)=\log\left(\frac{\frac{\pi_i}{\pi_i+\pi j}}{\frac{\pi_j}{\pi_i+\pi_j}}\right)=\log\left(\frac{\pi_i}{\pi_j}\right)=\beta_i-\beta_j$$

2.2 Model with Common Home Field Advantage

It is very common for home field advantage to play a significant role in determining the outcome of sporting events, so, not surprisingly, the Bradley-Terry model is often modified to account for contest venue.

Define P_{ij} such that i is always the home team with common home field advantage $\delta \in \mathbb{R}^+$ as a scaling parameter and $\log(\delta) = \alpha$.

$$P_{ij} = \frac{\delta\pi_i}{\delta\pi_i+\pi_j}$$

with log odds:

$$\log\left(\frac{P_{ij}}{1-P_{ij}}\right) = \beta_i - \beta_j + \alpha$$

Note that setting $\delta = 1$ results in the standard model.

2.3 Model with Individual Home Field Advantage

A model with individual home field advantage can be similarly obtained. If the amount of "boost" home field advantage provides varies by competitor, then this model may be more appropriate. The Bradley-Terry Model with individual home field advantage is specified as follows:

Define P_{ij} such that i is always the home team with individual home field advantage $\delta_i \in \mathbb{R}^+$ as a scaling parameter and $\log(\delta_i) = \alpha_i$.

with log odds:

$$\log\left(\frac{P_{ij}}{1-P_{ij}}\right) = \beta_i - \beta_j + \alpha_i$$

3. Likelihood Functions, Logistic Regression, and MLE Estimates

3.1 Likelihood Functions

There are many ways to parameterize the likelihood function for a Bradley-Terry Model. The following specification was chosen because it addresses all aspects of the model and uses a parameterization that is sensible for all stages of the modeling process:

Suppose there are N total games, $(i_1, j_1), (i_2, j_2)...(i_N, j_N)$, between K teams where each (i, j) pair represents a distinct pair of teams in $\{1, 2, ..., K\}$, and the home team is listed first in each pair.

Define Y_n, $(n = 1, 2, ..., N)$, such that:

$$Y_n = \begin{cases} 1 & \text{if home team } i \text{ defeats away team } j \text{ in the } nth \text{ game} \\ 0 & \text{if away team } j \text{ defeats home team } i \text{ in the } nth \text{ game} \end{cases}$$

The likelihood for parameters θ=$(\delta, \beta_2....\beta_K)$ is then given by:

$$L(\delta, \beta_2....\beta_K) = \prod_{n=1}^{N} \left[(p_{i_n j_n})^{Y_n} (1 - p_{i_n j_n})^{1-Y_n}\right] = \prod_{n=1}^{N} \left[\left(\frac{p_{i_n j_n}}{1-p_{i_n j_n}}\right)^{Y_n} (1 - p_{i_n j_n})\right]$$

where p_{ij} is given as a function of δ, β_i, β_j and is subject to the constraint $\beta_1 = 0$.

The Log-Likelihood is given by:

$$\begin{aligned} LogL(\alpha, \beta_2....\beta_k) &= \sum_{n=1}^{N} \left[Y_n log\left(\frac{p_{i_n j_n}}{1-p_{i_n j_n}}\right) + log(1 - p_{i_n j_n})\right] \\ &= \sum_{n=1}^{N} \left[Y_n(\alpha + \beta_{i_n} - \beta_{j_n}) - log(1 + e^{\alpha+\beta_{i_n}-\beta_{j_n}})\right] \end{aligned}$$

[8]

3.2 Logistic Regression

The equation

$$\log\left(\frac{P_{ij}}{1-P_{ij}}\right) = \beta_i - \beta_j + \alpha$$

from earlier actually specifies a simple logistic regression with a fairly unituitive setup.

Note that Y_n is a Bernoulli distributed random variable and the likelihood function takes the form of a generalized binomial. Because the generalized binomial is exponential family, $\text{logit}(P_{ij})=\log\left(\frac{Pij}{1-Pij}\right)$ can be used as a link-function relating the log-odds to a linear model. The logistic regression for the nth game in the model with common home field advantage is:

$$\text{logit}(P_{ij}|n)=X_{n1}\beta_1 + X_{n2}\beta_2, ..., +X_{nK}\beta_K + \alpha \ ,$$

where X_{nk} is defined as :

$$X_{nk} = \begin{cases} 1 & \text{if team } k \text{ at home in } nth \text{ game} \\ -1 & \text{if team } k \text{ away in } nth \text{ game} \\ 0 & \text{if team } k \text{ not playing in } nth \text{ game} \end{cases}$$

This results in the model specification; i.e., a column vector of the log-odds for all N games:

$$X\beta = \begin{bmatrix} X_{11} & X_{12} & ... & X_{1K} & 1 \\ X_{21} & X_{22} & ... & X_{2K} & 1 \\ \vdots & \vdots & ... & \vdots & \vdots \\ X_{N_1} & X_{N_2} & ... & X_{N_K} & 1 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_K \\ \alpha \end{bmatrix}$$

with design matrix dimensions N x $(K+1)$

3.3 MLE estimation equations

The MLE estimates can be obtained by first setting partial derivatives with respect to each parameter $(\alpha, \beta_2, ..., \beta_K)$ equal to 0, resulting in a system of $K+1$ non-linear equations in $K+1$ unknowns.

Although closed form solutions don't exist for the MLE estimates, they can easily be found provided a suitable algorithm is used. The most commonly used algorithms to perform the optimization are MM ("Majorize-Minimization" or "Minorize-Maximization") algorithms. This class of algorithms uses a surrogate function to find local minima and maxima of the objective function. In 1957, Ford showed that, for all MM algorithms, the optimization routine will converge to a unique maximum under certain conditions [2].

4 Model Adequacy

There are two conditions that must be satisfied in order for one of the standard variations of the Bradley-Terry Model to be appropriate:

- A linear ordering of the competitors by their strengths must exist
- For any two non-empty subsets of the competitors which partition the set of teams, S_1 and S_2, some team in S_1 has beaten some team in S_2 at least once. (The tournament graph is strongly connected) [7].

4.1 Bradley-Terry Assumption

The Bradley-Terry Model assumes each competitor has a "true strength" which is a fixed, positive number. Fixing the strengths implies that winning potential does not vary based on the opposing team or on the passage of time, opposite to what one might expect.

If this strength assumption is met, then a team's probability of winning should be transitive. This is to say if team A defeats team B and team B defeats team C, then team A should beat team C.

$$\text{If } p_{AB} > .5 \text{ and } p_{BC} > .5, \text{ then } p_{AC} > .5 \text{ and } p_{AC} > p_{BC}$$

The constant strength assumption fails if the winning probabilities are incoherent, for example, $p_{AB} > .5$ and $p_{BC} > .5$, and $p_{AC} < .5$. In this situation, the model will resolve this incoherence by assigning all the teams the same strengths. This results in predictions which are very inaccurate. Consequently, Bradley-Terry model is not appropriate for applications where the probabilities of winning are not transitive [7].

4.2 Condition for Convergence in Parameter Estimation and Interpretations

A graph theoretic interpretation of the conditions necessary for the existence of unique MLE estimators can be helpful with inutition, and also provides some useful results.

Define the tournament as a digraph G:$\{V(G), E(G)\}$ such that:

$V(G)$ is the set of vertices (which represent teams)

E(G) is the set of directed edges connecting vertices (which represent a competition)

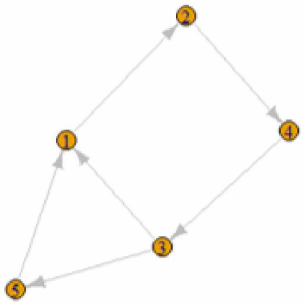
The "direction" in "directed graph ", refers to orientation of the edge (the way the arrow is pointing). Directed edges can be thought of as arrows pointing from winners to losers.

Note the following:

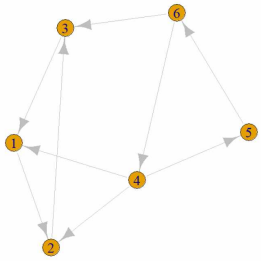
- A graph is connected if there is a path between every pair of vertices.
- A digraph is strongly connected if there exists a directed path from any vertex to any other vertex within the graph. Strong connectedness guarantees the existence of unique MLE estimates. Figure 1(a) shows an example of a strongly connected digraph.
- An underlying graph is the graph that results from ignoring the edge orientation of a digraph.
- A digraph is weakly connected if its' underlying graph is connected (A weakly connected digraph is a tournament where all teams have played each other at least once). Figure 1(b) shows an example of a graph that is weakly connected, but not strongly connected.
- A digraph is disconnected is there is a not path between every pair of vertices. Figure 1(c) shows a disconnected digraph (A disconnected digraph is a tournament where not all teams have played each other).
- A source is a vertex that only has outgoing edges (A source is a team that always wins). Figure 1(d) shows a digraph with a sink.
- A sink is a vertex that only has incoming edges (A sink is a team that always loses).

Consider the following digraphs:

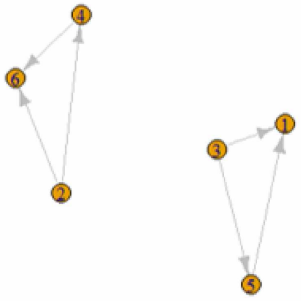
Figure 1
Example Digraphs



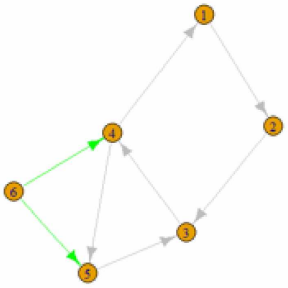
(a) Strongly-Connected Digraph
(MLE Estimates Exist)



(b) Digraph that is
weakly connected,
but not strongly connected
(MLE Estimates DNE)



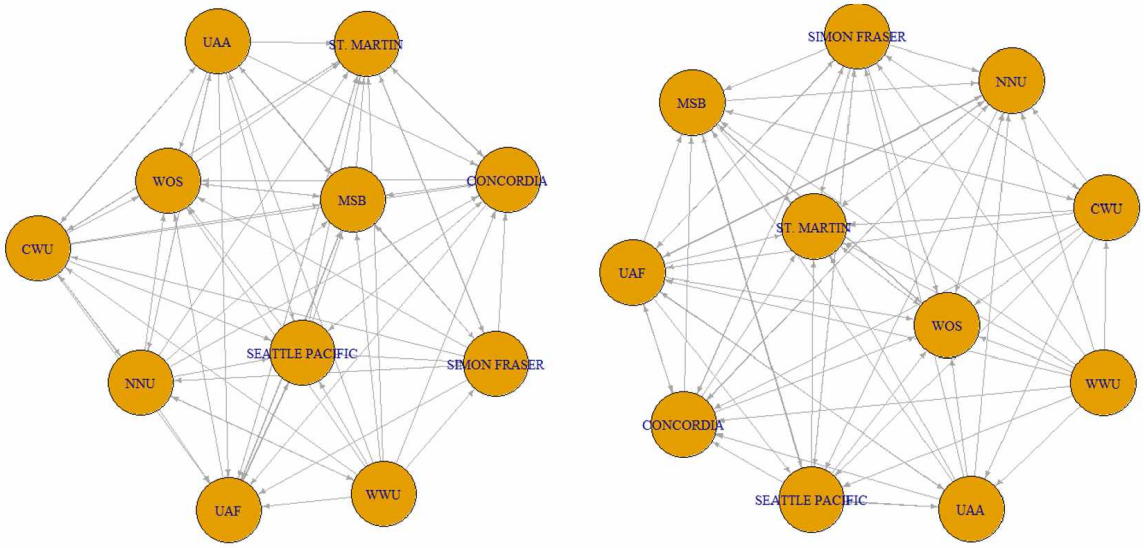
(c) Digraph with Disconnected
Subgraphs (MLE Estimates DNE)



(d) Digraph with Source
(MLE Estimates DNE)

The set-theory based criteria and equivalent strong connectedness guarantee the existence of unique MLE parameter estimates in very much the same way. They both ensure that there is a basis of comparison between any two teams either by way of a direct comparison or an intermediate outcome; both rule out tournaments where one subset of teams goes undefeated against a second partitioning subset. In these situations, the likelihood can always be increased by scaling the teams in the winning subset by a positive number greater than 1, so MLE parameter estimates do not exist [7].

Figure 2
2016-17 and 2018 Season Graphs



(a) 2016-2017 Training Data

(b) 2018 Test data

Above are the graphs for the training and test data. Note that the digraph for the training set is strongly connected, and therefore the MLE estimates exist for the MM algorithm used by BradleyTerry2. It is also worth pointing out that the graph for the test set is not strongly connected, because WWU didn't lose a single game in 2018.

5. Data Collection and Cleaning

The GNAC volleyball data was scraped using the python pandas package (code is included in the appendix) from the team websites. Most teams had a website where the data for all the years in a date range could be obtained by using a common URL and changing the year, which facilitated the scraping. However, there were many aspects of the extracted data that needed to be transformed after extraction:

- Team names had to be converted to a standard format. The data consisted of tables entered by 11 different teams over the course of three years, and many teams used their own unique naming conventions.
- Non-Conference Teams had to be filtered out. Each year each teams plays 20 in-conference games and a handful of out-of-conference games, with each GNAC team pair playing twice (once at home and once away). The out of conference games were omitted due to issues with sample size.
- There are three different Concordia Universities which regularly play games against teams in GNAC. Only the Concordia University in Portland is a member of GNAC. Concordia (Irvine) and Concordia (St. Paul) had to be filtered out leaving Concordia (Portland).
- Naming Conventions for UAF and UAA needed to be disambiguated. The used conventions for UAA and UAF were especially bothersome and non-standard. It took some time to figure out how to seperate them out properly.
- There was no variable for home/away (it had to be extracted from the Opponent Column). In the variable vs. opponent in the box score, home field is designated by presence/absence of at and vs. (e.g. UAF at UAA)
- Playoff and exhibition games had to be removed. Some teams choose to playoff games in their data sets, most included preseason games, and some included games against themselves.

After the data cleaning was completed, the games for 2016-2017 season were used as training data for the 2018 test set.

The data scraping and cleaning process took approximately 40 hours to complete.

6. Results

6.1 Model Fit

Table 1
2016-2017 GNAC Women's Volleyball Training Data

Table 1 shows the log-strength coeffficients for each of the eleven GNAC teams with UAF being used as a reference constraint. For the standard model, the log-strengths are relatively evenly distributed. The same cannot be said for the home field advantage models, which exhibit significant log-strength clustering. In situations where the strengths are close together, larger sample sizes are needed to determine dominance. This has implications that will be discussed in section 7.2.

*Some log-strengths are highlighted in red as examples of clustering.
*Detailed model descriptions along with home field advantage estimates are included in the appendix for each of the models.

Team	*Games Won 2016-2017*	*Results for Model I Standard Bradley-Terry*			*Results for Model II Bradley-Terry With Common Home-Field Advantage*			*Results for Model III Bradley-Terry With Individual Home-Field Advantage*		
		Log-Ability	*QSE*	*Rank*	*Log-Ability*	*QSE*	*Rank*	*Log-Ability*	*QSE*	*Rank*
Western Washington	36	4.72	0.57	1	5.19	0.79	1	4.63	0.77	1
UAA	33	4.08	0.49	2	4.49	0.73	2	4.51	0.78	2
Northwest Nazarene	29	3.40	0.43	3	3.73	0.68	3	2.78	0.63	3
Central Washington	27	3.00	0.42	4	3.30	0.67	4	2.48	0.63	4
Simon Fraser	26	2.96	0.43	5	3.26	0.67	5	2.14	0.63	5
Concordia	19	1.92	0.40	6	2.15	0.62	6	1.89	0.63	6
Seattle Pacific	17	1.59	0.40	7	1.69	0.61	7	0.94	0.68	7
Montana State Billings	10	0.51	0.43	8	0.57	0.59	8	0.00	0.72	9
Western Oregon	10	0.51	0.43	9	0.57	0.59	9	-0.33	0.77	10
UAF	7	0.00	0.48	10	0.00	0.00	10	-0.33	0.77	8
Saint Martin's	6	-0.19	0.50	11	-0.21	0.61	11	-1.68	1.14	11

Table 2
2018 GNAC Women's Volleyball Test Data and Model Prediction

Table 2 shows the number of actual wins vs. model predicted wins (where the model predicted a home team win with probability greater than or equal to .5) in the 2018 season for each of the models.

Note that the standard model predicts that no teams will have the same number of wins (although two pairs GNAC teams did have the same number of wins in 2018). Due to the reasons previously discussed, the home-field advantage models have numerous teams with the same ranking, making them

*The number of predicted wins highlighted in red represent duplicate rankings.

Team	*Games Won 2018*	*Results for Model I Standard Bradley-Terry*		*Results for Model II Bradley-Terry With Common Home-Field Advantage*		*Results for Model III Bradley-Terry With Individual Home-Field Advantage*	
		Number of Predicted Wins	*Rank*	*Number of Predicted Wins*	*Rank*	*Number of Predicted Wins*	*Rank*
Western Washington	20	20	1	19	1	16	1
Central Washington	16	13	3	14	4	16	1
UAA	14	16	2	18	2	16	1
UAF	12	6	8	3	8	1	11
Simon Fraser	12	12	4	14	4	15	4
Concordia	9	8	7	9	6	9	6
Seattle Pacific	9	9	6	9	6	9	6
Montana State Billings	7	5	9	3	8	5	8
Northwest Nazarene	6	11	5	15	3	10	5
Western Oregon	3	4	10	3	8	5	9
Saint Martin's	2	1	11	3	8	1	10

Model Accuracy	*71.80%*	*Model Accuracy*	*70.91%*	*Model Accuracy*	*69%*
Log-Loss	*1.896*	*Log-Loss*	*3.154*	*Log-Loss*	*3.134*
Null-Deviance	*303.6*	*Null-Deviance*	*303.6*	*Null-Deviance*	*303.6*
Residual-Deviance	*168.5*	*Residual-Deviance*	*155.33*	*Residual-Deviance*	*149.38*
Residual-Deviance DF	208	*Residual-Deviance DF*	209	*Residual-Deviance DF*	198
Average-Rank Misplacement(Predicted Wins)	*1.091*	*Average-Rank Misplacement(Predicted Wins)*	*1.636*	*Average-Rank Misplacement(Predicted Wins)*	1.456
Average-Rank Misplacement(Strength)	2.818	*Average-Rank Misplacement(Strength)*	2.818	*Average-Rank Misplacement(Strength)*	2.636

7. Discussion of Results

When looking at the appropriateness of a Bradley-Terry Model, the model must be assessed both by its ability to make valid predictions and produce correct rankings. This is done by employing tests for goodness-of-fit and empirical validity.

7.1 Residual Deviance Test

One of the main methods for assessing goodness-of-fit for the standard logistic regression is residual deviance testing. The residual deviance statistic is defined as:

$$D = -2\sum_{i \neq j} w_{ij} \log\left(\frac{\hat{p}_{ij}}{\frac{w_{ij}}{n_{ij}}}\right)$$

where w_{ij} is the number of times that team i beat team j in the n_{ij} games played between them. It can be shown that $D \sim \chi^2_{(C-K+1)}$, where C is the number of competitor pairs[7]. D is used as a test statistic to determine whether or not the fitted model is an improvement over the null model. The null hypothesis for this test is that the saturated model better fits the data than the null model.

Model	Residual Deviance	DF	P-value for H_0
I	168.5	209	.982
II	155.33	208	.997
III	149.39	198	.996

In order for the residual deviance test to be valid, in the case of a balanced design, there must be approximately 15 data points per cell [7]. Note that all models have P-values in excess of .1. However, in this data set, there are only 4 observations per cell. Therefore, the results for this test are inconclusive.

7.2 Systemic Testing

Although the residual deviance test isn't applicable for this particular situation, other ways to check model fit exist. The systemic test of the Bradley-Terry Model assesses transitivity by looking at paths up to some order, and comparing the relative incidences of coherence and incoherent paths. Systemic testing means that model fit can be checked using less data, because the dominance information being gathered isn't restricted to direct competitions. The R-package associated with the paper published on this subject was not functioning at the time that this paper was written, so it could not be implemented here [7].

7.3 Empirical Tests

Model accuracy can also be checked empirically by training the model on a subset of the overall data and then testing its predictive ability on the data that was held out. For this project, the simple metrics of accuracy and log-loss were used to assess the models trained on the 2016-2017 season and tested on the 2018 test data:

$$\text{Accuracy}=\frac{\text{Number of correct predictions}}{\text{Number of total predictions}}$$

$$\text{Log-loss}=-\frac{1}{N}\sum_{n=1}^{N} I(p_{i_n j_n}) \log p_{i_n j_n} \ ,$$

where $I(p_{i_n j_n})$ is an indicator function equal to one if the model prediction is correct and zero otherwise.

Table 3: Accuracy Metrics

Model	Accuracy
I	71.80%
II	70.91%
III	69.00%

Accuracy results for the all three models were around 70 percent, with the simplest model performing the best, followed by the models with individual-home-field-advantage and common homefield advantage, respectively.

Table 4: Log-Loss Metrics

Model	Log-Loss
I	-1.89
II	-3.15
III	-3.13

Again, the simplest model outperforms the two models with home field advantage in log-loss. However, this disparity in performance may be explained by the fact that the increased variability in strength in the more complicated models resulted in harsher penalties for making around the same number of bad predictions as the standard model.

7.4 Tests for Rank Agreement

The other objective of the Bradley-Terry model is to correctly predict rankings. There are a few different ways to measure ranking agreement; for the purposes of this paper, average-rank-agreement will be used.

$$\text{Average-Rank-Agreement} = \frac{1}{K} \sum |r_{actual} - r_{predicted}|$$

K=the number of teams in the tournament
r_{actual}=actual team ranking based on number of wins
$r_{predicted}$=predicted team ranking based on predicted number of wins

Table 5: Average Rank Agreement Metrics

Model	Average-Rank Misplacement(Predicted Wins)
I	1.09
II	1.63
III	1.46

Again, the standard model outperformed the home field advantage models in average rank agreement. The clustering in team strength strengths meant that on average, the home field advantage models rankings were off by approximately half of a rank more than the standard model rankings.

7.5 General Comments and Conclusions

It is worth observing the number of wins for each team by season. UAF and Northern Nazerene University stand out as having a large disparity between the number of wins in the training set and the number of wins in the test data. UAF averaged 3.5 wins in the 2016-2017 data and won 12 games in 2018. Northern Nazarene averaged 15 wins during the 2016-2017 seasons and won just 6 games in 2018. This could be for any number of reasons, such as player and coaching staff churn, program perception, recruiting, etc.

The versions of the Bradley-Terry model in this paper put equal weight on games used to create the model, regardless of when they occurred. Consequently, when predicting the outcome of a game played in 2018, the last game played in 2017 was as influential as the first game played in 2016. Models that accounted for the progression of time during seasons or that included a correction for the time elapsed between seasons were not implemented here for technical reasons, and using a smaller training set was not feasible because each team plays the other 10 teams just twice during the course of a season.

The closeness of team strengths and small sample size per cell mean that the transitivity assumption is very shaky. It seems that issues with the models' validity due to violations of the Bradley-Terry Assumption cannot be resolved within the framework of the unstructured model for this data set.

8 Recommendations For Future Work

Here are some ways that this project could be extended:

The unstructured model, which places no restrictions on the distribution of strength parameters, seems like an obvious weakness in this study. Fitting a Bayesian model to the data would help in a few different ways. First, it would allow for the model to be trained faster and deal with some of the issues surrounding sample size. Second, it would remove the strong-connnectedness requirement. Most importantly, it would allow for a model that evolves over time, possibly with a weighting scheme which puts more emphasis on newer games.

The methodology used to calculate the predicted number of wins, ($P_{i_n j_n} > .5$), here was purely deterministic. A more sound approach would have included something like sampling from a binomial distribution or have incorporated the binomial variance into the prediction in some other way.

Transitivity analysis of the model seems like it would be highly effective in dealing with some of the sample size issues associated with the residual deviance diagnostic test, and the framework for conducting this testing already exists in the Bradley-Terry Systemic Test. However, the Perc package associated with the systemic test has issues. If the systemic test were carried out and it showed that the transitivity of winning probabilites does not hold, then non-parametric dominance estimation could be carried out using conductance.

A simulation study could be used to study numerous facets of the model: the performance of Bayesian vs. unstructured variations of the model, transitivity analysis using systemic testing, and performance of path-based dominance estimation vs. the Bayesian and unstructured models.

References

[1] Raquel YS Aoki, Renato M Assuncao, and Pedro OS Vaz de Melo. Luck is Hard to Beat: The Difficulty of Sports Prediction. Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017.

[2] David R. Hunter. MM algorithms for generalized Bradley-Terry models. *Annals of Statistics*, 32:384–406, February 2004.

[3] Haziq Jamil. Analysis of paired comparison data using Bradley-Terry models with applications to football data . Master's thesis, The University of Warwick, 2010.

[4] Tao Jin, Pan Xu, Quanquan Gu, and Farzad Farnoud. Rank Aggregation via Heterogeneous Thurstone Preference Models. 2019.

[5] Franz J. Kiraly and Zhaozhi Qian. Modelling Competitive Sports: Bradley-Terry-Elo Models for Supervised and On-Line Learning of Paired Competition Outcomes. 2017.

[6] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[7] Aaron Shev, Kevin Fujii, Fushing Hsieh, and Brenda McCowan. Systemic Testing on Bradley-Terry Model against Nonlinear Ranking Hierarchy. *PLoS ONE 9(12): e115367*, 2014.

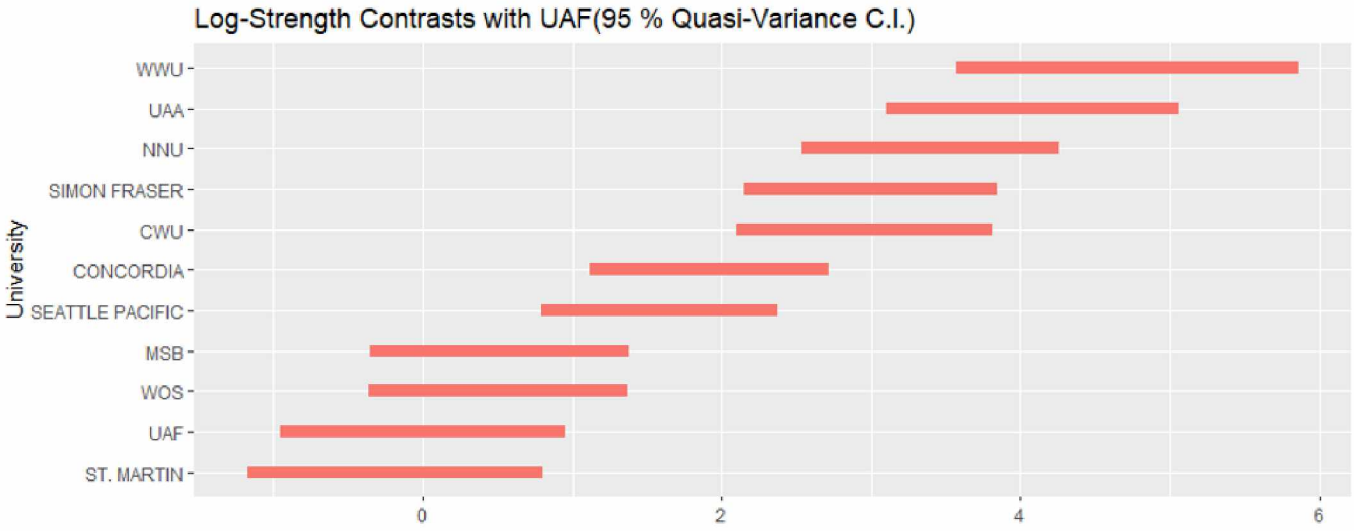
[8] Zhou Fan. Lecture 24 The Bradley-Terry model. 2016. `https://web.stanford.edu/class/archive/stats/stats200/stats200.1172/Lecture24.pdf`, Last accessed on 2019-10-8.

Appendix

Model I Results

Standard Bradley-Terry

Team	*Log-Strength*	*Std. Error*	*Quasi-Standard-Error*
Western Washington	4.72	0.79	0.57
Alaska Anchorage	4.08	0.73	0.49
Northwest Nazarene	3.40	0.68	0.43
Central Washington	3.00	0.67	0.42
Simon Fraser	2.96	0.67	0.43
Concordia	1.92	0.62	0.40
Seattle Pacific	1.59	0.61	0.40
Montana State Billings	0.51	0.59	0.43
Western Oregon	0.51	0.59	0.43
Alaska	0.00	0.00	0.48
Saint Martin's	-0.19	0.61	0.50

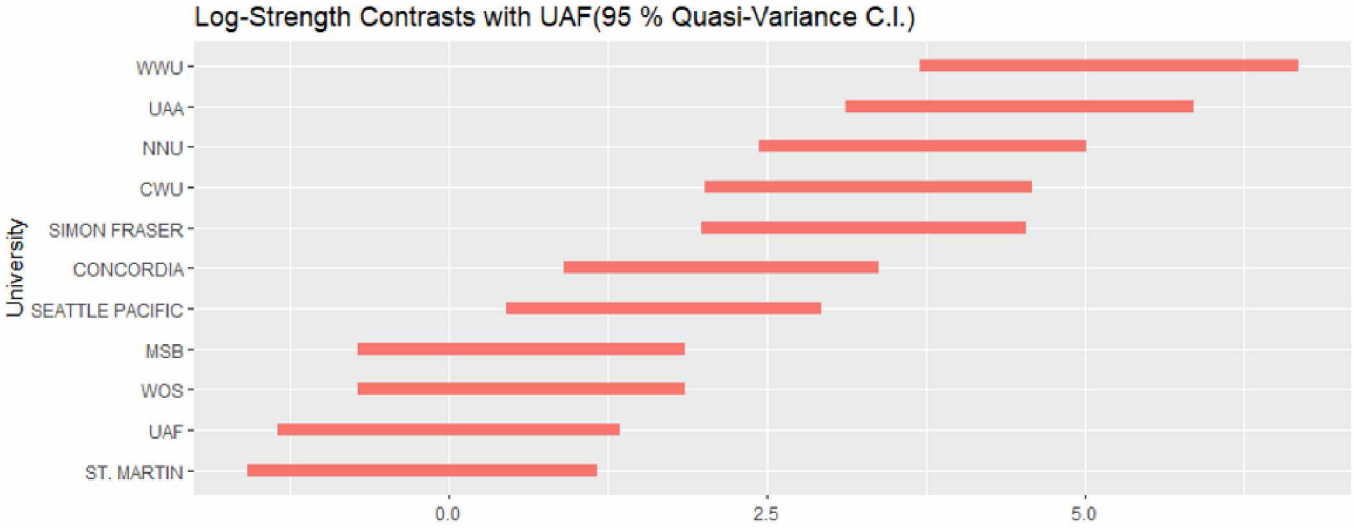


Model II Results

Bradley-Terry with Common Home-Field Advantage

University	*Log-Strength*	*Std. Error*	*Quasi-Standard-Error*	*95% CI Contrast with UAF*
WWU	5.19	0.79	0.57	(3.70, 6.68)
UAA	4.49	0.73	0.49	(3.13, 5.86)
NNU	3.73	0.68	0.43	(2.44, 5.01)
CWU	3.30	0.67	0.43	(2.02, 4.58)
SIMON FRASER	3.26	0.67	0.42	(1.99, 4.53)
CONCORDIA	2.15	0.62	0.40	(.9, 3.39)
SEATTLE PACIFIC	1.69	0.61	0.40	(.46, 2.93)
MSB	0.57	0.59	0.43	(-.72, 1.86)
WOS	0.57	0.59	0.43	(-.72, 1.85)
UAF	0.00	0.00	0.48	(-1.34, 1.34)
ST. MARTIN	-0.21	0.61	0.50	(-1.58, 1.16)

At-Home-Log-Advantage	
Common-Advantage	0.75

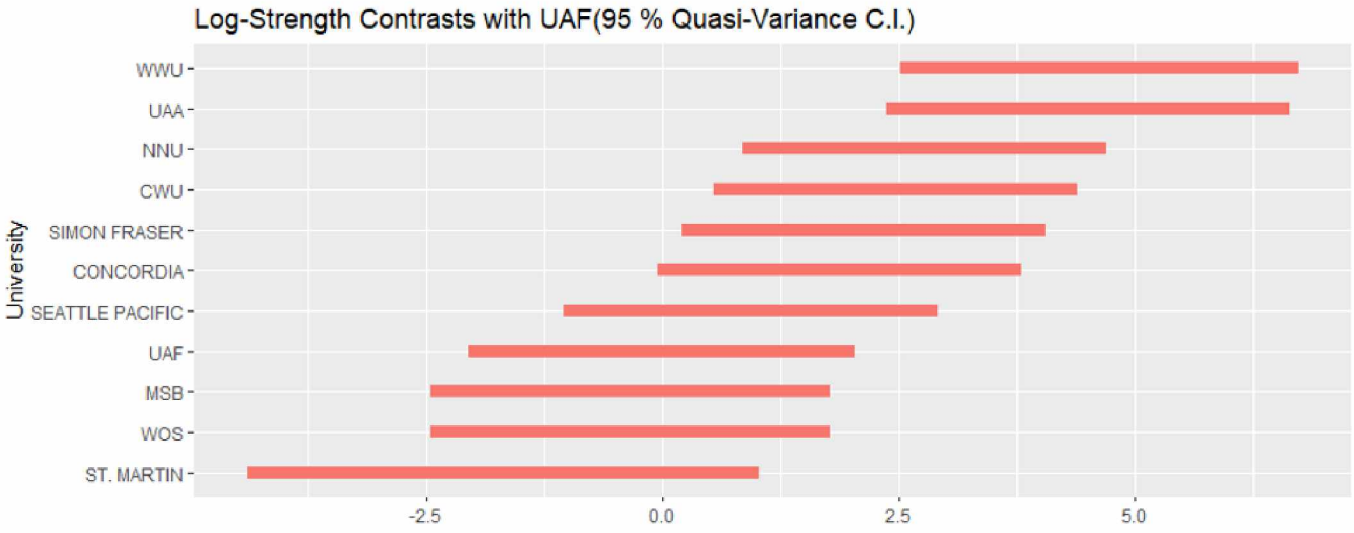


Model III Results

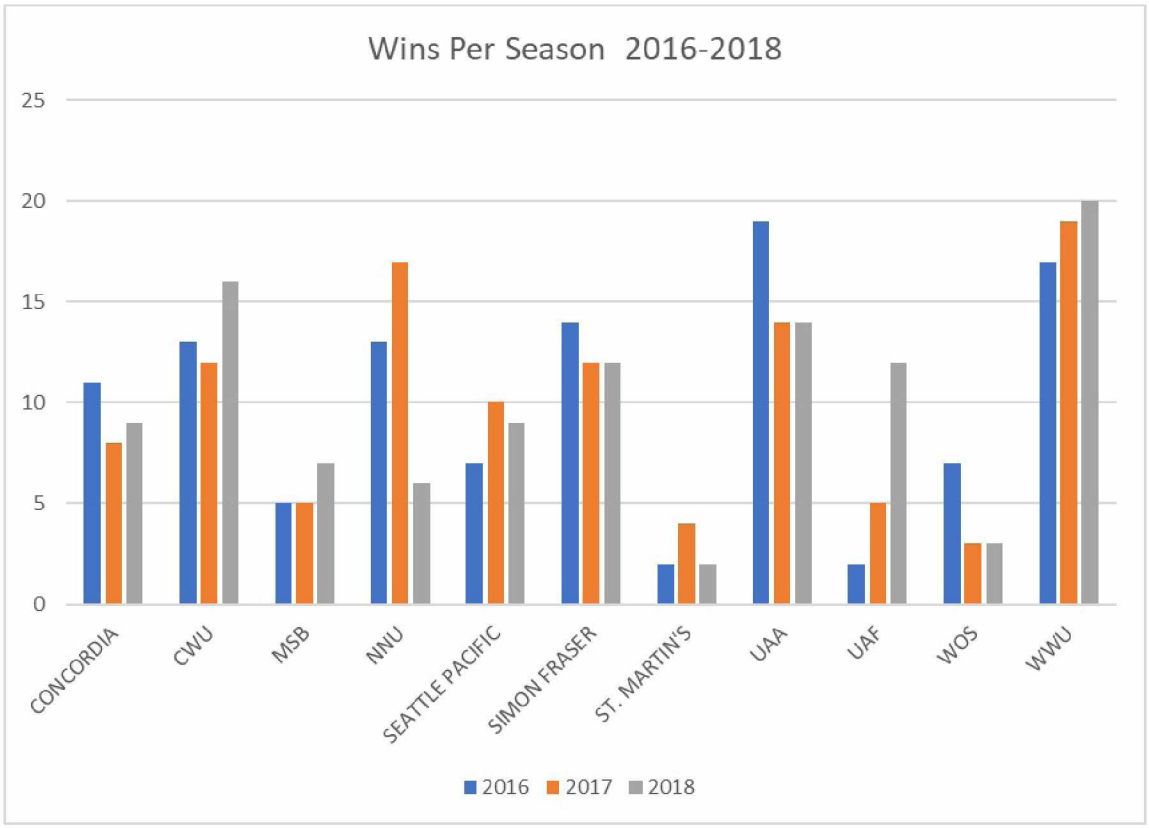
Bradley-Terry with Individual Home-Field Advantage

University	Log-Strength	Std. Error	Quasi-Standard-Error	95% CI Contrast with UAF
Western Washington	4.63	1.09	0.77	(2.52, 6.75)
UAA	4.51	1.09	0.78	(2.38, 6.64)
Northwest Nazarene	2.78	0.97	0.63	(.86, 4.70)
Central Washington	2.48	0.96	0.63	(.56, 4.40)
Simon Fraser	2.14	0.96	0.63	(.22, 4.07)
Concordia	1.89	0.96	0.63	(-.04, 3.81)
Seattle Pacific	0.94	0.97	0.68	(-1.04, 2.93)
UAF	0.00	0.00	0.72	(-2.05,2.05)
Montana State Billings	-0.33	1.04	0.77	(-2.45, 1.79)
Western Oregon	-0.33	1.04	0.77	(-2.45, 1.78)
Saint Martin's	-1.68	1.35	1.14	(-4.39, 1.03)

	At-Home-Log-Advantage	
University	Log-Advantage	SE
Saint Martin's	1.806	1.313
Simon Fraser	1.739	1.002
Northwest Nazarene	1.558	1.046
Central Washington	1.192	1.030
Western Oregon	1.056	1.006
Montana State Billings	1.040	1.006
Seattle Pacific	0.873	0.925
Western Washington	0.764	1.382
Concordia	-0.177	0.931
UAF	-0.613	1.061
UAA	-0.628	1.096



Other Data


Wins Per Season 2016-2018
25
20
15
10
5
0
CONCORDIA
CWU
MSB
NNU
SEATTLE PACIFIC
SIMON FRASER
ST. MARTIN'S
UAA
UAF
WOS
WWU
2016
2017
2018

Code

Data-Scraping

```
import pandas as pd
from urllib.request import Request, urlopen

#years for first type of url
years1=['2014-2015','2015-2016','2016-2017','2017-2018','2018-2019']

#years for second type of url
years2=['2014','2015','2016','2017','2018']

#type of year format associated with url
years=[years2,years2,years2,years2,years2,years2,years2,years2,years2,years2,
    ↪ years2,years2,years2,years2,years1,years1]

#first part of url
url1_list=['https://alaskananooks.com/cumestats.aspx?path=wvball&year=','https://
    ↪ alaskananooks.com/cumestats.aspx?path=wvball&year=','https://wildcatsports.
    ↪ com/cumestats.aspx?path=wvball&year=','https://wildcatsports.com/cumestats.
    ↪ aspx?path=wvball&year=','https://static.gocugo.com/custompages/Volleyball
    ↪ %20Stats/','https://static.gocugo.com/custompages/Volleyball%20Stats/','
    ↪ https://msubsports.com/cumestats.aspx?path=wvball&year=','https://
    ↪ msubsports.com/cumestats.aspx?path=wvball&year=','https://nnusports.com/
    ↪ cumestats.aspx?path=wvball&year=','https://nnusports.com/cumestats.aspx?
    ↪ path=wvball&year=', 'https://smusaints.com/cumestats.aspx?path=wvball&year=
    ↪ ','https://smusaints.com/cumestats.aspx?path=wvball&year=','https://
    ↪ athletics.sfu.ca/cumestats.aspx?path=wvball&year=','https://athletics.sfu.
    ↪ ca/cumestats.aspx?path=wvball&year=','https://static.wwuvikings.com/
    ↪ custompages/sports/w-volley/stats/','https://static.wwuvikings.com/
    ↪ custompages/sports/w-volley/stats/']

#second part of url
url2_list=['','','','','/teamgbg.htm#TGBG.TEM','/teamgbg.htm#TGBG.TEM','','','','
    ↪ ','','','','','/teamgbg.htm','/teamgbg.htm']

#position of table in website html array
table_num_list=[5,7,5,7,8,11,5,7,5,7,5,7,8,10,8,11]
```

Data-Scraping(Cont.)

```
#list of file outputs
output_list=['uaf.csv','cwu.csv','cwu_opponent.csv','concordia.csv','msb.csv','
    ↪ msb_opponent.csv', 'nnu.csv','stmartin.csv', 'simonfraser.csv','wwu.csv','
    ↪ montanastate.csv','centralwashington.csv','simonfraser.csv','uaf.csv','
    ↪ northwestnazarene.csv']

#search html tables to find position of box scores for season
def find_html(years, url1,url2, table_num,output):
    data=[0]*100
    schedule=[0]*100
    req = Request(url1+years[1]+url2, headers={'User-Agent': 'Mozilla/5.0'})
    webpage = urlopen(req).read()
    print(url1+years[1]+url2)
    data=pd.read_html(webpage)
    print(list(data[table_num]))

    return data[table_num]

#get data for the range for years given the table position
def get_html(years,url1,url2, table_num,output):
#initialize variables
    data=[0]*100
    schedule=[0]*100
#loop over years
    for x in range(len(years)):
        req = Request(url1+years[x]+url2, headers={'User-Agent': 'Mozilla/5.0'})
        print(url1+years[x]+url2)
        webpage = urlopen(req).read()
        #get html code for webpage
        data[x]=pd.read_html(webpage)
        #pull table from html
        schedule[x]=data[x][table_num]
        #give table a year value
        schedule[x]['Year']=years[x]
        print(years[x])

    #initialize aggregation sheet
    total=schedule[0]
    schedule=schedule[0:len(years)]

    #loop over schedule to create aggregation sheet
    for x in range(1,len(schedule)):
        total=total.append(schedule[x])
    total.to_csv(output)
```

Data-Cleaning(Cont.)

```
####################MANUAL EXCEL ADJUSTMENTS HERE##############################
#Create Spreadsheet with all years for all teams
tables=[0]*len(output_list)
for x in range(len(output_list)):
    tables[x]=pd.read_excel(output_list[x])
#various formatting issues
    tables[x].columns=[str.strip(str.upper(k)) for k in list(tables[x])]
    tables[x]['W/L']=tables[x]['W/L'].apply(lambda x: str.strip(x))
    tables['OPPONENT_VS']=tables.OPPONENT.apply(lambda x: str.upper(x))
    tables['OPPONENT']=tables.OPPONENT.apply(lambda x: str.upper(x))
    tables['UNIVERSITY']=tables.UNIVERSITY.apply(lambda x: str.upper(x))
#pull out variables of interest
    tables[x]=tables[x][['DATE', 'OPPONENT', 'W/L', 'SP', 'K', 'E', 'TA', 'PCT', '
        ↪ AST', 'SA', 'SE', 'RE', 'DIG', 'BS', 'BA', 'BE', 'TB', 'BHE', 'PTS', '
        ↪ YEAR', 'UNIVERSITY']]

#initialize main table
main_table=tables[0]
#concatenate all tables to main table
for x in range(1,len(output_list)):
    main_table=pd.concat([main_table,tables[x]],ignore_index=True)

working=main_table[:]

#make "dictionary" for variables
Fresno_Pacific=[['FRESNO'],[ 'Fresno␣Pacific']]
Montana_State=[['MSUB','BILLINGS'],['MSB']]
Northwest_Nazarene=[['NAZARENE'],['NNU']]
Saint_Martins=[['MARTIN'],["ST.␣MARTIN"]]
UAF=[['UAF', 'FAIRBANKS'],['UAF']]
UAA=[['UAA', 'ANCHORAGE'],['UAA']]
Simon_Fraser=[['FRASER'],['SIMON␣FRASER']]
CWU=[['CWU', 'CENTRAL␣WASHINGTON'],['CWU'],['CWU']]
CSMines=[['COLORADO␣MINES','COLO.␣SCH.␣OF␣MINES' ],['CSMines']]
CSEastBay=[['CAL␣STATE␣EAST␣BAY'],['CalStateEastBay']]
CSLA=[['CAL␣STATE␣L.A.','CAL␣STATE␣LA'],['CalStateLA']]
Hawaii=[['HAWAII␣PACIFIC'],['Hawaii␣Pacific']]
WOS=[['WESTERN␣ORE'],['WOS']]
Concordia=[['CONCORDIA'],['CONCORDIA']]
WWU=[['WESTERN␣WASHINGTON'],['WWU'],['WWU']]
Northern_State=[['NORTHERN␣STATE'],['Northern␣State']]
Seattle=[['SEATTLE'],['SEATTLE␣PACIFIC']]
```

Data-Cleaning(Cont.)

```
#GNAC Teams
Teams=[Montana_State,Northwest_Nazarene,Saint_Martins,UAF,Simon_Fraser,CWU,WOS,
    ↪ WWU,UAA,Seattle,Concordia]

#function to apply "dictionary"
def get_teams(x,Teams,flag):
    for team in Teams:
        for university_name in team[0]:
            if university_name in x:
                return team[1][0]

#modify variable using function
working['OPPONENT']=working['OPPONENT'].apply(lambda x: get_teams(x,Teams,False))


#filter out non-conference values which have been set to null
working2=working[working.OPPONENT.notnull()]
#filter out exhibition games
working2=working2[working2.UNIVERSITY!=working2.OPPONENT]
df=pd.read_excel('working2.xlsx')

#create ids to remove duplicate games
id=[0]*len(working2.index)
#create unique id using sorted teams and date
for x in range(len(working2.index)):
    id[x]=np.sort([working2.UNIVERSITY.iloc[x],working2.OPPONENT.iloc[x],str(
        ↪ working2.DATE.iloc[x])])
    id[x]="".join(id[x])


working2['id']=id

working2.reset_index()


working2=working2.drop_duplicates(subset=['id'])
```

Data-Formatting

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from scipy import stats
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
from patsy import dmatrices
from sklearn.preprocessing import StandardScaler

#readin data
df=pd.read_excel('C:\\Users\\daniel\\Desktop\\Volleyball\\Finished\\\\
    ↪ finalproject.xlsx')

#scaler for centering and normalizing data
scaler = StandardScaler()

#full variable list, reduced variable list, training years, test years
full_variable_set=['SP', 'K', 'E','TA', 'PCT','AST','SA','SE', 'RE','DIG','BS', '
    ↪ BA', 'BE','TB','BHE','PTS']

#dimension reduced variables
reduced_variable_set=['K','E','PCT','AST','RE','DIG','BHE','PTS']

#years for training the model
training_years=['2016','2017']

#year for testing the model
test_years=['2018']


#set parameters
def set_parameters(transform_list, training_year_list,labels,test_year_list):
    return

#call function
set_parameters(full_variable_set, reduced_variable_set,training_years,test_years)
```

Data-Formatting(Cont.)

```
#scale data, encode 'W/L', data sets for training, test, and aggregate
def preprocess(df,labels,training_year_list,test_year_list):
    global df1,df2,df3

    df[transform_list] = scaler.fit_transform(df[transform_list])

    df['W/L']=df['W/L'].apply(lambda x: float(x))

    df1=df[df['YEAR'].isin( training_year_list)]
    df2=df[df['YEAR'].isin(test_year_list)]

    df3=df[df['YEAR'].isin( training_year_list)]
    df3=df3.groupby('UNIVERSITY').aggregate('mean')

#call function
preprocess(df,transform_list,training_year_list,test_year_list)

#get game venue from vs string
def encode_location(x):

    #extract location from match string
    if 'AT' in str(x):
        return 0
    if 'VS' in str(x):
        return 2
    elif 'AT␣' not in str(x) and 'VS␣' not in str(x):
        return 1
```

Data-Formatting(Cont.)

```
def win_loss():
    global bt,bt2

    #create test and train tables
    bt,bt2=df1[:],df2[:]
    bt['Home.Win'],bt2['Home.Win']=0,0
    bt['Away.Win'],bt2['Away.Win']=0,0
    bt['Home'],bt2['Home']=0,0
    bt['Away'],bt2['Away']=0,0

    bt['home_away'],bt2['home_away']=bt.OPPONENT_VS.apply(lambda x:
        ↪ encode_location(x)),bt2.OPPONENT_VS.apply(lambda x: encode_location(x))
    bt['W/L'],bt2['W/L']=bt['W/L'].apply(lambda x: float(x)),bt2['W/L'].apply(
        ↪ lambda x: float(x))
    #drop games played on neutral ground
    bt,bt2=bt[bt.home_away.isin([0,1])],bt2[bt2.home_away.isin([0,1])]

    models=[bt,bt2]


    #initialize lists
    winner=[[],[]]
    loser=[[],[]]
    home_win=[[],[]]
    away_win=[[],[]]
    home_team=[[],[]]
    away_team=[[],[]]
```

Data-Formatting(Cont.)

```
#reformat data for test and train

#brute force assign variable values using loops and then merge lists with
    ↪ dataframe
for j in range(2):
    for i in range(len(models[j].DATE)):
        if models[j]['W/L'].iloc[i]==1:
            if models[j].home_away.iloc[i]==1:
                home_team[j].append(models[j].UNIVERSITY.iloc[i])
                away_team[j].append(models[j].OPPONENT.iloc[i])
                home_win[j].append(1)
                away_win[j].append(0)
                winner[j].append(models[j].UNIVERSITY.iloc[i])
                loser[j].append(models[j].OPPONENT.iloc[i])

            if models[j].home_away.iloc[i]==0:
                home_team[j].append(models[j].OPPONENT.iloc[i])
                away_team[j].append(models[j].UNIVERSITY.iloc[i])
                home_win[j].append(0)
                away_win[j].append(1)
                winner[j].append(models[j].UNIVERSITY.iloc[i])
                loser[j].append(models[j].OPPONENT.iloc[i])

        else:
            if models[j]['W/L'].iloc[i]==0:
                if models[j].home_away.iloc[i]==1:
                    home_team[j].append(models[j].UNIVERSITY.iloc[i])
                    away_team[j].append(models[j].OPPONENT.iloc[i])
                    home_win[j].append(0)
                    away_win[j].append(1)
                    loser[j].append(models[j].UNIVERSITY.iloc[i])
                    winner[j].append(models[j].OPPONENT.iloc[i])

                if models[j].home_away.iloc[i]==0:
                    home_team[j].append(models[j].OPPONENT.iloc[i])
                    away_team[j].append(models[j].UNIVERSITY.iloc[i])
                    home_win[j].append(1)
                    away_win[j].append(0)
                    loser[j].append(models[j].UNIVERSITY.iloc[i])
                    winner[j].append(models[j].OPPONENT.iloc[i])
```

Data-Formatting(Cont.)

```
    bt['home_team'], bt2['home_team']=home_team[0],home_team[1]
    bt['away_team'], bt2['away_team']=away_team[0],away_team[1]
    DATE=[bt.DATE,bt2.DATE]
    bt['Winner'],bt2['Winner']=winner[0],winner[1]
    bt['Loser'],bt2['Loser']=loser[0],loser[1]
    bt['home_win'],bt2['home_win']=home_win[0],home_win[1]
    bt['away_win'],bt2['away_win']=away_win[0],away_win[1]

    bt=pd.DataFrame(list(zip(home_team[0], away_team[0],home_win[0],away_win[0],
        ↪ DATE[0],bt.OPPONENT_VS,bt.id)),columns =['home_team','away_team','
        ↪ home_win','away_win','DATE','VS','id'])
    bt2=pd.DataFrame(list(zip(home_team[1], away_team[1],home_win[1],away_win[1],
        ↪ DATE[1],bt2.OPPONENT_VS,bt2.id)),columns =['home_team','away_team','
        ↪ home_win','away_win','DATE','VS','id'])

    #format data for R
    bt,bt2=bt.groupby(['home_team','away_team']).aggregate('sum'),bt2.groupby(['
        ↪ home_team','away_team']).aggregate('sum')
    bt,bt2=bt.reset_index(level='away_team'),bt2.reset_index(level='away_team')

    #output data to excel
    bt.to_excel('training')
    bt2.to_excel('testing')

win_loss()
```

R-Code for Fitting Model

```
bt<- read_excel('bt')

bt$home_team<-as.factor(bt$home_team)
bt$away_team<-as.factor(bt$away_team)

#change home and away team to factors
bt$home_team <- data.frame(team = bt$home_team,at.home=1)
bt$away_team <- data.frame(team = bt$away_team,at.home=0)
#home-away model specification
standard_model <- BTm(cbind(home_win, away_win), home_team, away_team,
                      data = bt, formula=~team, id = "team")

#common home-away model specification
home_away_model <- BTm(cbind(home_win, away_win), home_team, away_team,
                      data = bt, formula=~team+at.home, id = "team")

#readin modified spreadsheet with ones matrix for home-advantage factors
bt<- read_excel('bt')

#change home and away to factors
bt$home_team <- data.frame(team = bt$home_team, concordia=bt$concordia,cwu=bt$cwu
    ↪ ,msb=bt$msb,nn=bt$nn,stmartin=bt$stmartin,uaa=bt$uaa,uaf=bt$uaf,wos=bt$wos,
    ↪ wwu=bt$wwu,sf=bt$sf,seattle=bt$seattle,at.home=1)
bt$away_team <- data.frame(team = bt$away_team, concordia=0,cwu=0,msb=0,nn=0,
    ↪ stmartin=0,uaa=0,uaf=0,wos=0,wwu=0,sf=0,seattle=0,at.home=0)

#individual home-away model specification
individual_home_away_model <- BTm(cbind(home_win, away_win), home_team, away_team
    ↪ , data = bt, formula=~team +concordia+cwu+msb+nn+stmartin+uaa+uaf+wos+wwu+
    ↪ sf+seattle, id = "team")
```